ISSN 2186-5647

P-9

ゲート網羅故障検出率向上のための制御ポイント挿入法

日大生産工　○向後洋人　日大生産工(院)　青野竜弥　日大生産工　細川利典　　京産大　吉村正義

1.　はじめに

近年の超大規模集積回路(Very Large Integrated Circuit : VLSI)の大規模化・複雑化のため，製造中に物理的な欠陥が発生する可能性が高くなってきている．これらの欠陥を検出するために縮退故障や遷移故障などの故障モデルを用いた構造テストが必要である．しかしながら，上記の故障モデルですべての故障を検出できているにも関わらず，出荷された VLSI には不良 VLSI が存在する場合がある. 従来の信号線の故障を仮定した古典的な故障モデルを対象としたテストでは，セル内の欠陥を検出するには不十分である[2-4]．したがって，セル内の欠陥を対象にした新しい故障モデルとテスト手法が必要である．

これらの問題を解決する 1 つの手法として，ゲート網羅故障モデルとそのテスト生成法が提案されている[1]．ゲート網羅故障は，組合せ回路内のすべてのセルに着目したテスト生成法である．各セルの入力に対して各入力パターンの組合せを印加し，セルの出力の値を外部出力または擬似外部出力まで伝搬して観測することで各セルを網羅的にテストする．この方法は理論上，従来の故障モデルが見逃していた多くのセル内の欠陥を検出可能である．しかしながら，ゲート網羅故障モデルには縮退故障や遷移故障などに比べて故障検出率が低い[1][5]. 特定の入力パターンをセルに印加するための値の正当化や，セルの出力値を外部出力まで伝搬させる過程で，回路構造により不可能な値割当てが発生し，平均 84.32%の故障がテスト不能故障となる[9].

ある故障がテスト不能であるかどうかという問題はその回路情報やテスト条件を論理式で表現し，その式が充足可能か否かを判断する充足可能性問題（Satisfiability Problem : SAT）として定式化が可能である[6]．故障がテスト不能である場合，その論理式は充足不能（UNSAT）となる．この充足不能である根本原因を特定する技術の一つとして UNSAT コア[10]の抽出がある. UNSAT コアはテストを不可能にしている信号線の値割当ての最小限の集合を示すため，テスト不能の原因を抽出することが可能となる[7]．文献[10]では UNSAT コアに含まれているフリップフロップに対してスコアを加算して，最もスコアの高いフリップフロップを制御困難なフリップフロップとしている．

本論文では, UNSAT コアを用いてテスト不能故障の原因となっている回路内の論理値の衝突箇所を特定し，その状態を解消するために制御ポイントを挿入することで，テスト不能故障を検出可能にして，ゲート網羅故障モデルの検出率を向上させる手法を提案する．

本論文の構成は以下の通りである．第 2 章ではゲート網羅故障のテストやテスト生成手法について説明する．第 3 章では UNSAT コアの節数の削減手法について説明する．第 4 章では UNSAT コアを用いた信号線のスコア付けとそれに基づく制御ポイントの挿入手法を提案する．第 5 章で実験結果を示し，第 6 章では結論と今後の課題を述べる．

2.　ゲート網羅故障モデルとゲート網羅故障のテスト

本章では，ゲート網羅故障におけるテストと SAT を用いたテスト生成手法[9]について述べる．第 2.1 節ではゲート網羅故障のテストについて説明し，第 2.2 節では充足可能性(SAT)とゲート網羅故障のテスト生成モデル[9]について説明する．本論文ではセルとゲートは同じものとして扱う．

2.1.　ゲート網羅故障のテスト

ゲート網羅故障モデル[1]のテストでは，回路の各ゲートの入力に各値の組合せが印加され，ゲートの出力値を外部出力もしくは擬似外部出力まで伝搬し観測する．ゲート網羅故障の総数は回路内のゲート総数と各ゲートの入力数に比例する．回路のゲート網羅故障の総数 NF を以下に示す.

$$NF = \sum_{i=1}^{N} 2^{NI(g_i)}$$

N は回路のゲート数である．$NI(g_i)$はゲートg_iの入力数である．例として図 1 の例題回路 1 を用いて説明する．図 1 でのg_1からg_3はゲート，a から h は信号線である．ゲート数は 3 であり，g_1，g_2，g_3のそれぞれ入力数は 2，1，3 である．したがって，図 1 の回路のゲート網羅故障の総数は$2^2 + 2^1 + 2^3 = 14$である．図 1 の回路のゲート網羅故障の例を表 1 に示す．GF1 から GF3 はそれぞれg_1からg_3のゲート網羅故障集合を示している．

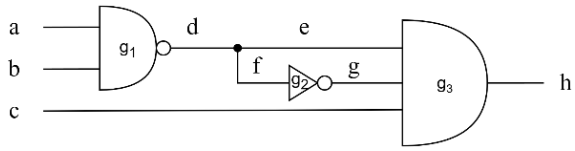

図 1. 例題回路 1

表 1. g_1，g_2，g_3のゲート網羅故障集合

GF1

ID	a	b	d
f_1	0	0	1/0
f_2	0	1	1/0
f_3	1	0	1/0
f_4	1	1	0/1

GF2

ID	f	g
f_5	0	1/0
f_6	1	0/1

GF3

ID	e	g	c	h
f_7	0	0	0	0/1
f_8	0	0	1	0/1
f_9	0	1	0	0/1
f_{10}	0	1	1	0/1
f_{11}	1	0	0	0/1
f_{12}	1	0	1	0/1
f_{13}	1	1	0	0/1
f_{14}	1	1	1	1/0

A Control Point Insertion Method to Improve Gate-Exhaustive Fault Coverage

Hiroto KOGO, Tatuya AONO, Toshinori HOSOKAWA and Masayoshi YOSHIMURA

2. 2. 充足可能性(SAT)とゲート網羅故障のテスト生成モデル

充足可能性問題(SAT)とはある命題論理式が与えられたとき，変数の値を定めることで式全体を真にすることが可能か否かを調べる問題である．これ以降で扱う SAT は特にブール充足可能性問題(Boolean Satisfiability Problem：SAT)であり，各リテラルは真(1)か偽(0)のいずれかの値をとるブール論理式に限定される．ほとんどの場合，SAT は論理積標準形(Conjunctive Normal Form：CNF)で表される命題論理式を入力とする．CNF は，リテラルの論理和である節をさらに論理積で結合した形式である．

一般に論理回路は命題論理式に変換可能である．例えば 2 入力 AND ゲート入力を X と Y，出力を Z とすると，命題論理式は$(x+\bar{z})\cdot(y+\bar{z})\cdot(\bar{x}+\bar{y}+z)$と表すことができる．他のすべてのゲートも同じように命題論理式に変換することができる．

論理回路のテスト生成も命題論理式に変換することができる[6]．各回路要素は入出力伝搬規則を用いて，いくつかの節で表現することができる．故障を含まない正常回路と故障の推移的ファンアウト[9]のみの回路である故障回路を命題論理式として追加することにより，正常回路と故障回路の違いを命題論理式で表現することができる．ゲート網羅故障のゲートへの入力値割当てや故障値割当ては単位節として追加する．その後，SAT ソルバーを使用することにより，充足可能ならば式全体を充足する(擬似)外部入力に対応する変数割当てを探索し，充足不能ならばその故障がテスト不能であることが証明される．図 2 に図 1 の回路のg_1に$(a,b,d)=(1,1,0/1)$の故障がある際の具体的な正常回路，故障回路，故障検出回路(EXOR)の例を示す．単位節として i が真になる制約を追加して SAT ソルバーを用いて，この故障に対してテスト可能か否かの判定が可能である．

また，SAT 分野の近年の進歩により[8]，命題論理式が充足不可能となる原因の節の集合を特定することが可能である．もとの命題論理式の節の部分集合であり，かつ充足不能である節の集合を UNSAT コア[7]という．その UNSAT コアの中でも，1 つでも節を削除すると SAT になるような UNSAT コアを最小 UNSAT コアという．本論文の中で扱う UNSAT コアはすべて最小 UNSAT コアを指す．

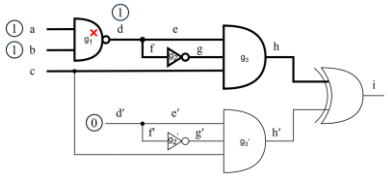

図 2. ゲート網羅故障のテスト生成モデル例

3. UNSAT コアを用いた回路内の衝突箇所の探索

本章では UNSAT コアの節数の削減について説明する．UNSAT コアはもとの命題論理式の部分集合であるという性質上，大規模な部分集合となる傾向があるため，ある回路の故障の命題論理式が UNSAT になったとしても，その UNSAT コアからテスト不能の原因を直接探索するのは困難である．よって UNSAT コアの節数を削減する必要がある．そのために，論理回路で実行する含意操作の結果を命題論理式に追加する．含意操作の結果を追加することにより節の削減が可能である．これは SAT ソルバーの中で実行している学習節を生成する処理と類似している．$(\bar{a}+b)\cdot(a)$という命題論理式を例に説明を行う．この論理式を真にするためには a は必ず真でなければならない．そのため，$(\bar{a}+b)$という節から(b)を導きだすことができ，その結果$(\bar{a}+b)$という節は削除し，$(a)\cdot(b)$というような節の削除と追加により SAT ソルバーは SAT を解いている．これを含意操作の結果を追加することにより実現可能である．例を図 3 の例題回路 2 を用いて含意操作の追加例と UNSAT コアの削減の例を説明する．図 3 のg_1，g_2はゲート，a から d までは信号線である．図 4 に示した回路はテスト不能故障である回路上のゲート g_2 の入力 0 のゲート網羅故障$(g_2,0,1/0)$に関するテスト生成モデルである．なお，本論文では故障を(セルのインスタンス名，入力値，出力値)という形式で記述する．ゲートg_2の入力が 0 である故障のため，信号線 d には 0 を割当て，信号線 b' には 0 を割当てる．図 4 のテスト生成モデルの CNF を以下に示す．

$(e+\bar{c})\cdot(b+\bar{c})\cdot(\bar{e}+\bar{b}+c)\cdot(\bar{d}+\bar{b})\cdot(d+b)\cdot(\bar{a}+e)\cdot(a+\bar{e})\cdot(\bar{a}+d)\cdot(a+\bar{d})\cdot(e+\bar{c'})\cdot(b'+\bar{c'})\cdot(\bar{e}+\bar{b'}+c')\cdot(\bar{c}+\bar{c'}+\bar{f})\cdot(c+c'+\bar{f})\cdot(c+\bar{c'}+f)\cdot(\bar{c}+c'+f)\cdot(f)\cdot(\bar{d})\cdot(\bar{b'})$

上記 CNF の UNSAT コアを以下に示す．

$(e+\bar{c})\cdot(a+\bar{e})\cdot(\bar{a}+d)\cdot(b'+\bar{c'})\cdot(c+c'+\bar{f})\cdot(f)\cdot(\bar{d})\cdot(\bar{b'})$

例題回路 2 に対して前方含意操作と後方含意操作を行うことにより，信号線 a を 0，信号線 b を 1 と割当て可能である．含意操作の結果$a=0$を追加することで，UNSAT コアから$(\bar{a}+d)$と$(\bar{d})$を削減することができる．含意操作追加後の UNSAT コアを以下に示す．

$(e+\bar{c})\cdot(a+\bar{e})\cdot(b'+\bar{c'})\cdot(c+c'+\bar{f})\cdot(f)\cdot(\bar{b'})\cdot(\bar{a})$

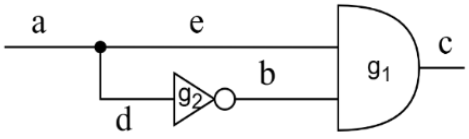

図 3. 例題回路 2

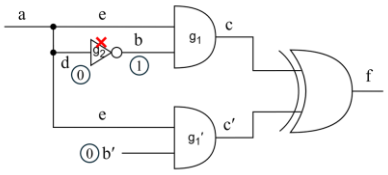

図 4. 例題回路 2 のg_2故障モデル

4. ゲート網羅故障のための UNSAT コアを用いた制御ポイントの挿入手法

本章ではゲート網羅故障を検出するための制御ポイント挿入箇所の選択手法を提案する．4.1 節に UNSAT コアを用いたスコア付けの手法について述べ, 4.2 節に制御ポイントの挿入手法について述べる．最後に全体アルゴリズムを示す.

4. 1. UNSAT コアを用いたスコア付け

UNSAT コアにはテスト不能の原因が含まれている可能性が高い．その中でも，複数の UNSAT コアに含まれているゲートは特にテスト不能の原因が含まれている可能性が高い[7]．テスト不能の原因はいずれかのゲートの入力値の正当化が不可能であることだと考えられるため, UNSAT コアに節が含まれる正常回路のゲートの入力信号線にスコアを加算することで制御ポイントの挿入箇所を絞り込む.

4. 2. 制御ポイントの挿入

スコア付けを全テスト不能故障数分行ったあと，制御ポイントの挿入箇所を決定する．回路内で再収斂構造となっている信号線の値を制御可能にすることによりテスト不能の原因を改善可能であると考えられるため MUX を最もスコアが高い分岐先信号線に挿入する.

全テスト不能故障に対してスコア付けを実行した例を図 5 の例題回路 3 を用いて示す．aからkは信号線である．テスト不能故障数は 9 であり，$(g_1, 00, 1/0)$, $(g_1, 01, 1/0)$，$(g_1, 10, 1/0)$，$(g_1, 11, 1/0)$，$(g_2, 1, 0/1)$，$(g_3, 00, 0/1)$，$(g_3, 11, 0/1)$，$(g_5, 10, 1/0)$, $(g_5, 11, 1/0)$ である．スコア付けの例を故障$(g_1, 00, 1/0)$を用いて説明する．この故障の UNSAT コアを抽出し，削減を行った結果を以下に示す.

$$(i + \bar{f}) \cdot (f + h + \bar{\imath}) \cdot (h)$$

今回の例ではg_5の入力信号線hとfにスコアを加算する．全テスト不能故障によるスコア加算例を図 6 に示す．外部入出力信号線への挿入を不可能にするためスコアは-1 としている．最後に分岐先信号線に挿入するため，図 6 の例では信号線 k に挿入する.

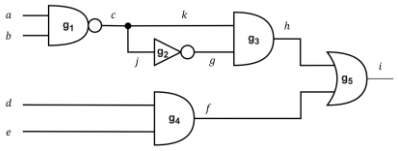

図 5. 例題回路 3

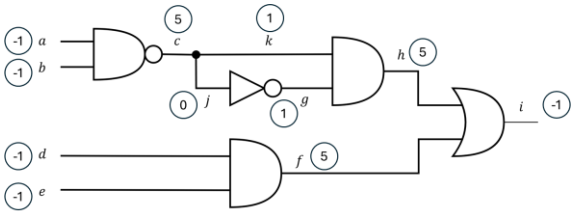

図 6. 例題回路 3 のスコア付け結果

4. 3. 全体アルゴリズム

本節では，提案手法であるテスト不能なゲート網羅故障を検出するための UNSAT コアを用いた制御ポイント挿入箇所選択手法のアルゴリズムを述べる．下記の図 7 に本手法の全体アルゴリズムを示す．入力は回路 C である．出力は制御ポイント挿入により向上した故障検出率 FC である．また，RED_SET はテスト不能故障集合, num_inserted_MUX は挿入制御ポイント数, num_sig は全信号線数，CNF は生成したテスト生成モデル CNF，CNF_added は CNF に含意操作の結果を追加した CNF，UNSAT_CORE は CNF から抽出した UNSAT コア，TARGET_SIG は制御ポイント挿入対象信号線である.

はじめに，回路からテスト不能故障集合を作成する(1 行目)．3 行目から 10 行目まで，全信号線数の 2%に制御ポイントを挿入するまで処理を行う(2 行目)．4 行目から 7 行目まで，全テスト不能故障に対してスコア付け処理を行う(3 行目). まずは選択したテスト不能故障に対してテスト生成モデル CNF を生成する(4 行目). 次に生成した CNF に含意操作の結果を追加した CNF_added を作成する(5 行目)．次にその CNF から UNSAT コアを抽出する(6 行目)．最後にその UNSAT コアを基にスコア付けを行う(7 行目). すべてのテスト不能故障に対してスコア付け処理を実行し，分岐先信号線の中でスコアが最大の信号線 TARGET_SIG を選択する(9 行目). 次に選択した信号線に対して制御ポイントを挿入する(10 行目). 全信号線の 2%に制御ポイント挿入後，最後に向上した故障検出率 FC を計算する(12 行目)．故障検出率 FC を返却する(13 行目).

```
Algorithm  Control Point Insert for Gate Exhaustive Fault Model
Input    :  Circuit C
Output   :  Improved fault coverage FC
1.  RED_SET = mk_UNSAT_fault_set ( C ) ;
2.  while( num_inserted_MUX < num_sig * 0.02)then
3.  | while(RED_SET ≠ Φ)then
4.  | | CNF = mk_CNF_from_RED_FAULT ( RED_SET ) ;
5.  | | CNF _added= add_implication_CNF ( RED_SET, CNF );
6.  | | UNSAT_CORE = Extraction_UNSAT Core (CNF_added) ;
7.  | | SIG_SCORE = score_from_UNSAT Core ( UNSAT_CORE ) ;
8.  | endwhile
9.  | TARGET_SIG = calc_max_score();
10. | insert_MUX( TARGET_SIG );
11. endwhile
12. FC = calc_FC();
13. return( FC ) ;
14. end
```

図 7. 制御ポイント挿入場所選択アルゴリズム

5. 実験結果

提案手法は C 言語で実装し，ISCAS’89 ベンチマーク回路に対して，Intel Gold 5320(2.20GHz)および 128GB メモリを搭載したコンピュータを用いて実験を行った. SAT ソルバーは CaDiCaL のバージョン 2.1.3 を使用した．挿入した制御ポイント数は全ての回路において全信号線数の 2%である．

表 2 に回路の基本情報を示す．それぞれ，circuit は回路名，all はゲート網羅故障数，det はテスト可能故障数，red はテスト不能故障数，FC は故障検出率を示す．本手法では，故障検出率が 90%以下の回路を実験対象とした．

表 3 に実験結果を示す．それぞれ circuit は回路名，mux は挿入制御ポイント数, dec_f は制御ポイント挿入により検出可能となったテスト不能故障数，+FC は制御ポイント挿入により向上した故障検出率の差分，dec/red はテスト不能故障のうちテスト可能となったテスト不能故障数の割合，TIME は本アルゴリズムの実行時間を示している．表 3 から，全信号線数の 2%の制御ポイント数で, 故障検出率を平均で 4.7%向上させ，テスト不能故障の平均 33.0%をテスト可能故障にした．実験結果は UNSAT コアによるスコア付けが概ね有効であることを示している．

表 2．回路の基本情報

	all	det	red	FC
s298	652	566	86	86.8%
s5378	9530	8135	1395	85.4%
s9234	17148	14372	3052	83.8%
s13207	24604	21552	3052	87.6%
s15850	28700	25481	3219	88.8%

表 3．実験結果

circuit	mux	dec_f	+FC	dec/red	TIME[s]
s298	5	20	3.1%	23.3%	0.12
s5378	105	238	2.5%	17.1%	370.55
s9234	184	1611	9.4%	52.8%	1727.66
s13207	262	1603	6.5%	52.5%	6365.59
s15850	316	619	2.2%	19.2%	30686.34

6. まとめ

本論文では UNSAT コアを用いた故障検出率向上のための制御ポイント挿入法を提案した. ISCAS’89 ベンチマーク回路の実験結果は故障検出率を平均 4.7%向上させることができた．今後の課題として，制御ポイントごとに設定されている制御信号線を 1 本にまとめること，値割当不可能なゲート網羅故障を用いたより精巧なスコア付け手法の提案などが挙げられる．

7. 参考文献

[1] Kyoung Youn Cho, S. Mitra and E. J. McCluskey, "Gate exhaustive testing," *IEEE International Conference on Test, 2005.*, Austin, TX, USA, 2005, pp. 7 pp.-777.

[2] Z. Liu, B. Niewenhuis, S. Mittal and R. D. Blanton, "Achieving 100% cell-aware coverage by design," *2016 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, Dresden, Germany, 2016, pp. 109-114.

[3] F. Hapke *et al.*, "Cell-Aware Test," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 9, pp. 1396-1409, Sept. 2014.

[4] F. Hapke, J. Schloeffel, H. Hashempour and S. Eichenberger, "Gate-Exhaustive and Cell-Aware pattern sets for industrial designs," *Proceedings of 2011 International Symposium on VLSI Design, Automation and Test*, Hsinchu, Taiwan, 2011, pp. 1-4.

[5] R. Guo, S. Mitra, E. Amyeen, J. Lee, S. Sivaraj and S. Venkataraman, "Evaluation of test metrics: stuck-at, bridge coverage estimate and gate exhaustive," *24th IEEE VLSI Test Symposium*, Berkeley, CA, USA, 2006, pp. 6 pp.-71.

[6] T. Larrabee, "Test pattern generation using Boolean satisfiability," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 11, no. 1, pp. 4-15, Jan. 1992.

[7] Andre Suelflow, Goerschwin Fey, Roderick Bloem, and Rolf Drechsler. 2008. “Using errors,”in Proceedings of the 18th ACM Great Lakes symposium on VLSI (GLSVLSI '08). New York, NY, USA, 77–82.

[8] Lintao Zhang and S. Malik, "Validating SAT solvers using an independent resolution-based checker: practical implementations and other applications," *2003 Design, Automation and Test in Europe Conference and Exhibition*, Munich, Germany, 2003, pp. 880-885.

[9] R. Asami, T. Hosokawa, M. Yoshimura and M. Arai, "A Multiple Target Test Generation Method for Gate-Exhaustive Faults to Reduce the Number of Test Patterns Using Partial MaxSAT," *2020 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, Frascati, Italy, 2020, pp. 1-6.

[10] K. J. Balakrishnan and L. Fang, "RTL Test Point Insertion to Reduce Delay Test Volume," *25th IEEE VLSI Test Symposium (VTS'07)*, Berkeley, CA, USA, 2007, pp. 325-332.